

BAB II

LANDASAN TEORI

Pada bab ini akan dijelaskan landasan teori yang terkait dengan permasalahan yang dibahas dan juga menjelaskan sistem yang digunakan pada Tugas Akhir ini. Adapun landasan teori yang digunakan sebagai berikut:

2.1 Sistem *Dashboard*

2.1.1 Visualisasi

Menurut Frey (2008:4), sebuah visualisasi yang tepat adalah semacam narasi yang memberikan jawaban jelas atas pertanyaan tanpa rincian asing. Dengan berfokus pada tujuan awal dari pertanyaan, Anda dapat menghilangkan rincian seperti itu karena pertanyaan itu memberikan acuan untuk apa yang diperlukan dan apa yang tidak diperlukan.

2.1.2 Pengertian *Dashboard*

Dashboard adalah sebuah tampilan visual dari informasi terpenting yang dibutuhkan untuk mencapai satu atau lebih tujuan, digabungkan dan diatur pada sebuah layar, menjadi informasi yang dibutuhkan dan dapat dilihat secara sekilas. *Dashboard* itu sebuah tampilan pada satu monitor komputer penuh yang berisi informasi yang bersifat kritis, agar kita dapat mengetahui hal-hal yang perlu diketahui. Biasanya kombinasi teks dan grafik, tetapi lebih ditekankan pada grafik (Few, 2006:34).

2.1.3 Tujuan Penggunaan *Dashboard*

Tujuan penggunaan *dashboard* menurut Eckerson (2006:5) yaitu:

1. Mengkomunikasikan Strategi

Mengkomunikasikan strategi dan tujuan yang dibuat oleh eksekutif kepada semua pihak yang berkepentingan sesuai dengan peran dan levelnya dalam organisasi.

2. Memonitor dan Menyesuaikan Pelaksanaan Strategi

Memonitor pelaksanaan dari rencana dan strategi yang telah dibuat. Memungkinkan eksekutif untuk mengidentifikasi permasalahan kritis dan membuat strategi untuk mengatasinya.

3. Menyampaikan Wawasan dan Informasi ke Semua Pihak

Menyajikan informasi menggunakan grafik, simbol, bagan dan warna yang memudahkan pengguna dalam memahami dan mempersepsi informasi secara benar.

2.1.4 Jenis *Dashboard*

Dashboard bisa dikelompokkan sesuai dengan level manajemen yang didukungnya menurut Eckerson dan Few dalam (Hariyanti 2008:10) yaitu:

1. *Operational Dashboard*

- a. Mendukung manajemen level operasional.
- b. Memberikan informasi tentang aktivitas yang sedang terjadi, beserta perubahannya secara *real time* untuk memberikan kewaspadaan terhadap hal-hal yang perlu direspon secara cepat.
- c. Fokus pada *monitoring* aktifitas dan kejadian yang berubah secara konstan.

- d. Informasi disajikan spesifik, tingkat kedetailan yang cukup dalam.
- e. Media penyajian yang sederhana.
- f. *Alert* disajikan dengan cara yang mudah dipahami dan mampu menarik perhatian pengguna.
- g. Bersifat dinamis, sehingga memerlukan data *realtime*.
- h. Didesain untuk berinteraksi dengan data, untuk mendapatkan informasi yang lebih detail, maupun informasi pada level lebih atas (*HigherLevel Data*).

2. *Tactical Dashboard*

- a. Mendukung manajemen *tactical*.
- b. Memberikan informasi yang diperlukan oleh analisis untuk mengetahui penyebab suatu kejadian.
- c. Fokus pada analisis untuk menemukan penyebab dari suatu kondisi atau kejadian tertentu.
- d. Dengan fungsi *drilldown* dan navigasi yang baik.
- e. Memiliki konten informasi yang lebih banyak (Analisis perbandingan, pola atau tren, evaluasi kerja).
- f. Menggunakan media penyajian yang “cerdas” yang memungkinkan pengguna melakukan analisis terhadap data yang kompleks.
- g. Didesain untuk berinteraksi dengan data.
- h. Tidak memerlukan data *realtime*.

3. *Strategic Dashboard*

- a. Mendukung manajemen level strategis.

- b. Informasi untuk membuat keputusan bisnis, memprediksi peluang, dan memberikan arahan pencapaian tujuan strategis.
- c. Fokus pada pengukuran kinerja *high-level* dan pencapaian tujuan strategis organisasi.
- d. Mengadopsi konsep *Balance Score Card*.
- e. Informasi yang disajikan tidak terlalu detail.
- f. Konten informasi tidak terlalu banyak dan disajikan secara ringkas.
- g. Informasi disajikan dengan mekanisme yang sederhana, melalui tampilan yang *unidirectional*.
- h. Tidak di desain untuk berinteraksi dalam melakukan analisis yang lebih detail.
- i. Tidak memerlukan data *real time*.

2.1.5 Karakteristik Dashboard

Karakteristik *dashboard* menurut (Eckerson, 2006) yaitu:

1. Model pemrosesan berdasarkan kejadian yaitu menangkap kejadian setiap saat dari beberapa sistem yang mencakup dan mempengaruhi proses bisnis.
2. Aturan bisnis yang kuat yaitu mengizinkan penggunaannya membuat peringatan, target, ambang untuk menilai kinerja individu.
3. *Dashboard* bisnis yang *user friendly* yaitu memperbarui nilai sebagai aliran kejadian melalui sistem dan menempatkan nilai tersebut dalam hubungan dengan menghubungkan ke pencapaian bisnis.
4. Sebuah sistem aliran kerja yang bergabung dan bekerjasama yang mengizinkan penggunaannya untuk memulai proses secara formal dan informal, yang dengan proses itu pengguna dapat berkolaborasi mendiskusikan hasilnya

Beberapa karakteristik *dashboard* menurut Malik (Hariyanti, 2008:8)

yaitu:

1. Sinergi

Ergonomis dan memiliki tampilan visual yang mudah dipahami oleh pengguna. *Dashboard* mensinergikan informasi dari berbagai aspek yang berbeda dalam satu layar.

2. Monitor

Menampilkan KPI yang diperlukan dalam pembuatan keputusan dalam domain tertentu, sesuai dengan tujuan pembangunan *dashboard* tersebut.

3. Akurat

Informasi yang disajikan harus akurat, dengan tujuan untuk mendapatkan kepercayaan dari penggunanya.

4. Responsif

Merespon *threshold* yang telah didefinisikan, dengan memberikan *alert* (seperti bunyi alarm, *blinker*, email) untuk mendapatkan perhatian pengguna terhadap hal-hal yang kritis.

5. *Timely*

Menampilkan informasi terkini yang diperlukan untuk pengambilan keputusan.

6. Interaktif

Pengguna dapat melakukan *drilldown* dan mendapatkan informasi lebih detail, analisis sebab akibat dan sebagainya.

7. *More Data History*

Melihat tren sejarah KPI contohnya perbandingan jumlah mahasiswa baru saat ini dengan beberapa tahun yang lalu, untuk mengetahui apakah kondisi sekarang lebih baik atau tidak.

8. *Personalized*

Penyajian informasi spesifik untuk setiap jenis pengguna sesuai domain tanggung jawab, hak akses dan batasan akses data.

9. *Analytical*

Fasilitas untuk melakukan analisis seperti sebab akibat.

10. *Collaborative*

Fasilitas pertukaran catatan laporan antar pengguna mengenai hasil pengamatan *dashboard*-nya masing-masing yaitu sarana komunikasi dalam melakukan fungsi manajemen dan control.

11. *Trackability*

Memungkinkan setiap pengguna untuk mengkustomisasi nilai yang akan dilacak.

2.1.6 **Komponen Dashboard**

Dalam memahami perbedaan diantara ketiga jenis *dashboard* kinerja, perlu untuk mengetahui masing-masing komponen aplikasi yang digunakan. Meskipun tidak ada aturan keras dan cepat tentang penggunaan komponen, Gambar 2.1 yang memberikan beberapa pedoman umum (Eckerson, 2006:106).

PERFORMANCE DASHBOARD COMPONENTS			
	Operational Dashboard	Tactical Dashboard	Strategic Dashboard
Monitoring	Dashboard	BI Portal	Scorecard
Analysis	Statistical models Decision engines	OLAP analysis Interactive reporting Advanced visualization Scenario modeling	Time-series analysis Standard reports
Management	Alerts Agents	Workflow Usage monitoring Auditing	Meetings Annotations Strategy maps

The monitoring, analysis, and management components that most commonly comprise the three types of performance dashboards.

Gambar 2.1 Komponen *Dashboard*

1. Komponen *Dashboard* Operasional

Dashboard operasional menggunakan antarmuka *dashboard* untuk memantau proses operasional. *Dashboard* memberikan peringatan yang memberitahukan pengguna tentang kondisi pengecualian dalam proses yang sedang mereka pantau sehingga mereka dapat bertindak cepat untuk memperbaiki masalah atau memanfaatkan peluang.

2. Komponen *Dashboard* Taktis

Dashboard taktis sering menampilkan hasil dalam *business intelligence* (BI) portal yang berisi grafik dan tabel serta dokumen lainnya pengguna perlu untuk memantau proyek atau proses yang mereka kelola. Portal ini dibangun ke sebagian besar alat BI dan biasanya mengintegrasikan dengan portal komersial yang banyak digunakan perusahaan untuk menjalankan *intranet* perusahaan mereka.

3. Komponen *Dashboard* Strategis

Dashboard Strategis menggunakan antarmuka *scorecard* untuk melacak kinerja terhadap tujuan strategis. Meskipun mereka mirip dengan antarmuka *dashboard*, *scorecard* umumnya melacak kemajuan kelompok secara bulanan daripada secara tepat waktu. *Scorecard* umumnya menampilkan lebih metrik seluruh spektrum yang lebih luas dari organisasi daripada *dashboard*, terutama di *scorecard* perusahaan. Informasi kinerja dalam antarmuka *scorecard* biasanya lebih diringkas dari dalam antarmuka *dashboard*.

2.1.7 Kesalahan Umum Pembuatan *Dashboard*

Beberapa hal dibawah ini merupakan 13 kesalahan umum pada pembuatan *dashboard* (Few, 2006):

1. Melebihi batas pada satu layar monitor komputer. Hal ini mengacu pada tampilan *dashboard*.
2. Menyediakan data yang tidak memadai: misal *dashboard* tentang penerimaan mahasiswa baru, seharusnya *dashboard* yang ada tidak hanya berisi jumlah mahasiswa baru pada tahun itu saja, melainkan berisi informasi jumlah mahasiswa baru tahun lalu.
3. Menampilkan detil atau presisi yang berlebihan: *dashboard* hampir selalu memerlukan informasi tingkat tinggi untuk mampu mendukung penggunaanya untuk peninjauan cepat. Jadi dengan detil yang berlebihan, hanya akan memperlambat penangkapan si pengguna tanpa menambah keuntungan pengguna. Contoh: \$3.8M akan lebih baik dibanding \$3.848.352,93.

4. Memilih ukuran kurang tepat: misalnya, bila seorang pengguna *dashboard* hanya memerlukan persentase tingkat penjualan, maka sebaiknya hanya disajikan dalam bentuk persentase (-9% akan lebih baik dibanding -\$8.066)
5. Memilih media tampilan yang tidak tepat: maksudnya adalah salah memilih media (bar, pie, circle, atau radar).
6. Menyajikan variasi berbeda yang sia-sia: misalnya, menyajikan chart penjualan pada beberapa daerah dengan menggunakan pie, radar, dan bar pada *dashboard* yang sama.
7. Menggunakan media tampil yang desainnya payah.
8. Menampilkan kuantitas data secara tidak akurat: contoh sebuah grafik batang yang dimulai angka \$500.000 bukan \$0.
9. Mengatur tampilan data dengan payah. *Dashboard* pada dasarnya menampilkan informasi yang banyak dengan tampilan seminimalis mungkin. Jadi, bila data yang ada tidak diatur sedemikian rupa, akan semakin membingungkan penggunanya.
10. Menyoroti data penting secara tidak efektif atau tidak sama sekali. *Dashboard* yang baik adalah menonjolkan data yang lebih penting dibanding yang lain. Sehingga pengguna langsung melihatnya.
11. Mengacaukan tampilan dengan dekorasi yang tak perlu. Sebaiknya tampilan *dashboard* tidak terlalu “wah” tampilannya, hal ini akan menyebabkan mata penggunanya mudah lelah di kemudian hari.
12. Salah atau berlebihan menggunakan warna. Sebaiknya menggunakan warna yang tepat. Dan tidak serampangan dalam menggunakan warna.
13. Mendesain tampilan yang tidak atraktif. Misalnya tidak ada comboboxnya.

2.1.8 Key Performance Indikator

Key Performance Indicator adalah indikator yang merepresentasikan kinerja dari proses yang dilaksanakan (Hariyanti, 2008). *Key Performance Indicator* merupakan sekumpulan ukuran mengenai aspek kinerja yang paling kritis, yang menentukan kesuksesan organisasi pada masa sekarang dan masa yang akan datang. *Key Performance Indicator* digunakan memprediksi peluang kesuksesan atau kegagalan dari proses-proses yang dilaksanakan organisasi, sehingga KPI dapat digunakan sebagai alat untuk meningkatkan kinerja organisasi secara dramatis. Contoh dari penjelasan diatas adalah penentuan parameter nilai dalam trend penerimaan mahasiswa baru membuat user dapat dengan mudah mengetahui kondisi penerimaan mahasiswa baru apakah sedang bagus atau tidak.

2.1.9 Grafik

Ada beberapa tipe grafik atau diagram yang dapat digunakan untuk menampilkan gambaran informasi supaya lebih jelas, antara lain (Santosa, 1994):

1. Diagram garis

Diagram garis digunakan untuk menunjukkan perubahan nilai dari sederetan data relatif terhadap waktu, karena diagram garis biasanya digunakan untuk menunjukkan suatu kecenderungan atau *trend*.

2. Diagram batang

Diagram batang digunakan untuk menyajikan nilai relatif terhadap data yang lain. Misal, eksekutif ingin melihat grafik pendaftar per-tahun dan per-gelombang.

3. Diagram roti (pie)

Diagram pie biasanya digunakan untuk menggambarkan besarnya prosentase data. Misalkan menggambarkan besarnya prosentase alasan mahasiswa keluar.

Grafik dapat digunakan untuk menunjukkan keterhubungan antar data, seperti perbandingan nominal, *time-series*, deviasi, korelasi, dan sebagainya . Ada berbagai macam bentuk grafik yang dapat dipilih untuk menggambarkan setiap jenis keterhubungan data, seperti yang terdapat pada tabel 2.1. Namun demikian, grafik kurang bisa menampilkan angka dengan format yang presisi.

Tabel 2.1. Keterhubungan Data dan Jenis Grafik yang Sesuai (Hariyanti, 2008).

No	Keterhubungan Data	Jenis Grafik yang sesuai
1	Perbandingan nominal	a. Grafik bar (horisontal atau vertikal) b. Grafik titik (jika 0 tidak termasuk dalam skala nilai)
2	<i>Time-series</i>	c. Grafik garis (untuk melihat tren seluruh data) d. Grafik bar (untuk melihat perbandingan antar nilai individu) e. Grafik titik yang dihubungkan dengan garis (untuk melihat nilai individu sekaligus tren data secara keseluruhan)
3	Ranking	f. Grafik bar (horisontal atau vertikal) g. Grafik titik (jika 0 tidak termasuk dalam skala nilai)
4	Bagian dari keseluruhan	h. Grafik bar (horisontal maupun vertikal) i. Grafik stack bar j. Pie chart
5	Deviasi	k. Grafik garis l. Grafik titik yang dihubungkan dengan garis
6	Distribusi frekuensi	m. Grafik bar vertikal/histogram (untuk menunjukkan nilai individu) n. Grafik garis/poligon frekuensi (untuk menunjuk tren data secara keseluruhan)

No	Keterhubungan Data	Jenis Grafik yang sesuai
7	Korelasi	o. Grafik titik dan garis (scatter-plot)

2.2 Analisis dan Perancangan Sistem

Menurut Kendall dan Kendall (2003:7), analisis dan perancangan sistem dipergunakan untuk menganalisis, merancang, dan mengimplementasikan peningkatan-peningkatan fungsi bisnis yang dapat dicapai melalui penggunaan sistem informasi terkomputerisasi. Analisis sistem dilakukan dengan tujuan untuk dapat mengidentifikasi dan mengevaluasi permasalahan yang terjadi dan kebutuhan yang diharapkan, sehingga dapat diusulkan perbaikannya.

Perancangan sistem merupakan penguraian suatu sistem informasi yang utuh ke dalam bagian komputerisasi yang dimaksud, mengidentifikasi dan mengevaluasi permasalahan, menentukan kriteria, menghitung konsistensi terhadap kriteria yang ada, serta mendapatkan hasil atau tujuan dari masalah tersebut serta mengimplementasikan seluruh kebutuhan operasional dalam membangun aplikasi.

2.3 *Unified Modeling Language*

Menurut Nugroho (2005:16), pemodelan *visual* adalah proses penggambaran informasi-informasi secara grafis dengan notasi-notasi baku yang telah disepakati sebelumnya. Notasi-notasi baku sangat penting demi suatu alasan komunikasi. Dengan notasi-notasi pemodelan yang bersifat baku komunikasi yang baik akan terjalin dengan mudah antar anggota tim pengembang sistem/perangkat

lunak dan antara anggota tim pengembang dengan para pengguna. Untuk melakukan pemodelan sistem/perangkat lunak, dalam buku ini notasi-notasi *Unified Modeling Language* (UML) yang akan digambarkan secara elektronik (dengan bantuan komputer) lewat sarana perangkat lunak. Dengan pemodelan menggunakan UML ini, pengembang dapat melakukan:

1. Tinjauan umum bagaimana arsitektur sistem secara keseluruhan.
2. Penelaahan bagaimana objek-objek dalam sistem saling mengirim pesan (*message*) dan saling bekerjasama satu sama lain.
3. Menguji apakah sistem/perangkat lunak sudah berfungsi seperti yang seharusnya.
4. Dokumentasi sistem/perangkat lunak untuk keperluan-keperluan tertentu di masa yang akan datang.

2.4 *Database*

Menurut Marlinda (2004:1), *database* adalah suatu susunan/kumpulan data operasional lengkap dari suatu organisasi/perusahaan yang diorganisir/dikelola dan disimpan secara terintegrasi dengan menggunakan metode tertentu menggunakan komputer sehingga mampu menyediakan informasi optimal yang diperlukan pemakainya. Penyusunan satu *database* digunakan untuk mengatasi masalah-masalah pada penyusunan data yaitu redundansi dan inkonsistensi data, kesulitan pengaksesan data, isolasi data untuk standarisasi, banyak pemakai (*multiple user*), masalah keamanan (*security*), masalah kesatuan (*integration*), dan masalah kebebasan data (*data independence*).

2.5 HTML5

Menurut Ikhsan, dkk (2012:4) HTML5 merupakan standar baru untuk HTML, XHTML dan DOM HTML. Sejak munculnya HTML versi 4.01, perkembangan website semakin berkembang. Pada versi terbaru ini HTML5 menyajikan beberapa kelebihan-kelebihan, diantaranya :

1. Dapat ditulis dalam sintaks HTML (dengan tipe media txt/html) dan XML.
2. Intigeritas yang lebih baik dengan aplikasi web dan pemrosessannya.
3. Penulisan kode yang lebih efisien.

2.6 *Hypertext Preprocessor*

Menurut Firdaus (2007:2), PHP merupakan singkatan dari *Hypertext Preprocessor*, adalah sebuah bahasa *scripting* berbasis *server side scripting* yang terpasang pada HTML dan berada di *server* dieksekusi di *server* dan digunakan untuk membuat halaman *web* yang dinamis. Sebagian besar sintaksnya mirip dengan bahasa C atau java, ditambah dengan beberapa fungsi PHP yang spesifik. Tujuan utama bahasa ini adalah untuk memungkinkan perancang *web* menulis halaman *web* dinamis dengan cepat.

Halaman *web* biasanya disusun dari kode-kode HTML yang disimpan dalam sebuah *file* berekstensi .html. *File* HTML ini dikirimkan oleh *server* (atau *file*) ke *browser*, kemudian *browser* menerjemahkan kode-kode tersebut sehingga menghasilkan suatu tampilan yang indah. Lain halnya dengan program PHP, program ini harus diterjemahkan oleh *web server* sehingga menghasilkan kode html yang dikirim ke *browser* agar dapat ditampilkan. Program ini dapat berdiri sendiri ataupun disisipkan di antara kode-kode HTML sehingga dapat langsung

ditampilkan bersama dengan kode-kode HTML tersebut. Program php dapat ditambahkan dengan mengapit program tersebut di antara tanda `<? dan ?>`. Tanda-tanda tersebut biasanya digunakan untuk memisahkan kode php dari kode HTML. *File* HTML yang telah dibubuhi program php harus diganti ekstensi-nya menjadi `.php` atau `.php3`.

2.7 MySQL

MySQL adalah *database* yang menghubungkan *script* PHP menggunakan perintah *query* dan *escape character* yang sama dengan PHP. PHP memang mendukung banyak *database*, tetapi untuk membuat sebuah *web* yang dinamis selalu *Up to Date*, MySQL merupakan pilihan *database* tercepat saat ini (Firdaus, 2007:3).

MySQL (*My Structured Query Language*) atau yang bisa dibaca mai-sekuel adalah program pembuat dan pengelola *database*. Selain itu data Mysql juga merupakan program pengakses *database* yang bersifat jaringan, sehingga dapat digunakan untuk Aplikasi *Multi User* (banyak pengguna). Kelebihan dari MySQL adalah menggunakan bahasa *query* (permintaan) standar SQL (*Structured Query Language*). SQL adalah suatu bahasa permintaan yang terstruktur.

2.8 JavaScript

Menurut Hakim (2010:2), *javascript* merupakan bahasa *scripting* yang dapat bekerja di sebagian besar *web browser*. *Javascript* dapat disisipkan didalam *web* menggunakan *tag script*. *Javascript* dapat digunakan untuk banyak tujuan, misalnya untuk membuat efek *roolover* baik gambar maupun *text*, dan untuk

membuat *AJAX Javascript* adalah bahasa yang digunakan untuk *AJAX*. Kode *java script* juga dapat diletakkan di *file* tersendiri yang berekstensi *javascript (.js)*. *Script* tersebut akan dieksekusi ketika dipanggil berdasarkan *trigger* pada *event* tertentu.

2.9 *Highcharts*

Highcharts adalah *library* pembuatan *chart* yang ditulis dalam *JavaScript* murni, menawarkan cara mudah untuk menambahkan grafik interaktif ke situs *web* atau aplikasi *web*. *Highcharts* saat ini mendukung *line, spline, area, area spline, column, bar, pie, scatter, angular gauges, area range, area spline range, column range, bubble, box plot, error bars, funnel, waterfall* dan *polar chart types*.

